CLAIMS:

1. (Currently Amended): A byte execution unit, comprising:

logic coupled to receive byte instruction information ~~and two operands~~, to receive a first operand from a first source register, to receive a second operand from a second source register, and configured to perform an operation specified by the byte instruction information upon at least one of the ~~two operands~~ first operand or the second operand, thereby producing a result in a destination register, wherein the byte instruction information specifies either a count ones in bytes operation, an average bytes operation, an absolute differences of bytes operation, or a sum bytes into halfwords operation,

wherein responsive to a count ones in bytes operation, the byte execution unit counts a number of logical one bits in each byte of at least the first operand and stores each result in a corresponding byte of the destination register;

wherein responsive to an average bytes operation, the byte execution unit averages each byte of the first operand with a corresponding byte of the second operand and stores each result in a corresponding byte of the destination register;

wherein responsive to an absolute differences of bytes operation, the byte execution unit subtracts each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, determines an absolute value of each intermediate result to form a final result, and stores each final result in a corresponding byte of the destination register; and

wherein responsive to a sum bytes into halfwords operation, the byte execution unit sums a number of corresponding one-byte portions of the first operand or the second operand and stores each result in a corresponding halfword of the destination register.

2. (Currently Amended): The byte execution unit as recited in claim 1, wherein each [[of]] the ~~two operands~~ first operand and the second operand comprises a plurality of bits, and wherein the bits of each of the ~~two operands~~ first operand and the second operand are grouped to form a plurality of corresponding 8-bit bytes.

3. (Currently Amended): The byte execution unit as recited in claim 2, wherein each of the ~~two operands~~ first operand and the second operand comprises 128 bits, and wherein the bits of each of the ~~two operands~~ first operand and the second operand are grouped to form 16 corresponding bytes.

4-7. (Canceled)

8. (Currently Amended): A byte execution unit, comprising:

pre-processing logic coupled to receive a plurality of operands and configured to perform an operation upon the operands dependent upon an operation specified by a byte instruction, thereby producing an intermediate result;

adder logic coupled to receive the intermediate result and configured to perform an addition operation upon the intermediate result, thereby producing a sum and a sum+1; [[and]]

post-processing logic coupled to receive the sum and sum+1 and configured to perform an operation upon the sum and sum+1 dependent upon the operation specified by a byte instruction, thereby producing a result; and

a control unit coupled to the pre-processing logic, the adder logic, and the post-processing logic,

wherein responsive to a count ones in bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to count a number of logical one bits in each byte of at least the first operand and to store each result in a corresponding byte of the destination register;

wherein responsive to an average bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to average each byte of the first operand with a corresponding byte of the second operand and to store each result in a corresponding byte of the destination register;

wherein responsive to an absolute differences of bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to subtract each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, to determine an absolute value of each

intermediate result to form a final result, and to store each final result in a corresponding byte of the destination register; and

wherein responsive to a sum bytes into halfwords operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to sum a number of corresponding one-byte portions of the first operand or the second operand and to store each result in a corresponding halfword of the destination register.

9. (Canceled)

10. (Canceled)

11. (Original): The byte execution unit as recited in claim 8, wherein the pre-processing logic comprises population counter logic coupled to receive the operands and configured to produce population output signals indicative of numbers of logic ones in portions of the operands.

12. (Original): The byte execution unit as recited in claim 8, wherein the pre-processing logic comprises compressor logic coupled to receive the operands and configured to perform a compression function.

13. (Original): The byte execution unit as recited in claim 8, wherein the post-processing logic comprises end-around carry logic configured to perform an end-around carry function.

14. (Original): The byte execution unit as recited in claim 8, wherein the post-processing logic is configured to perform bit shift operations.

15-18. (Canceled)

19. (Original): A byte execution unit, comprising:

a plurality of byte units, wherein each byte unit comprises:

a plurality of population counters each coupled to receive a portion of a first operand and configured to produce a population output signal indicative of a number of logic ones in the corresponding portion of the first operand;

a first compressor unit coupled to receive a portion of the first operand and configured to produce a first plurality of compressor output signals dependent upon the first operand;

a second compressor unit coupled to receive a portion of the second operand and configured to produce a second plurality of compressor output signals dependent upon the second operand;

adder input multiplexer logic coupled to receive the population output signals and the first and second pluralities of compressor output signals as data input signals, and a first plurality of control signals, and configured to produce a portion of the data input signals as output signals dependent upon the first plurality of control signals;

adder logic coupled to receive the output signals produced by the adder input multiplexer logic and configured to produce a plurality of adder output signals dependent upon the output signals produced by the adder input multiplexer logic; and

result multiplexer logic coupled to receive the adder output signals as data input signals, and a second plurality of control signals, and configured to produce a portion of the data input signals as a result signal dependent upon the second plurality of control signals;

wherein the byte execution unit is coupled to receive byte instruction information, and wherein the first and second pluralities of control signals are indicative of the byte instruction information, and wherein the byte instruction information specifies either a count ones in bytes operation, an average bytes operation, an absolute differences of bytes operation, or a sum bytes into halfwords operation.

20. (Original): The byte execution unit as recited in claim 19, wherein the first and second operands each comprise a plurality of bits, and wherein the bits of the first and second operands are grouped to form a plurality of corresponding 8-bit bytes.

21. (Original): The byte execution unit as recited in claim 20, wherein the first and second operands each comprise 128 bits, and wherein the bits of the first and second operands are grouped to form 16 corresponding bytes.

22. (Original): The byte execution unit as recited in claim 20, wherein in the event the byte instruction information specifies the count ones in bytes operation, the result signal is indicative of a number of logic one bits in each of the bytes of the first operand.

23. (Original): The byte execution unit as recited in claim 20, wherein in the event the byte instruction information specifies the average bytes operation, the result signal is indicative of averages of corresponding bytes of the first and second operands.

24. (Original): The byte execution unit as recited in claim 20, wherein in the event the byte instruction information specifies the absolute differences of bytes operation, the result signal is indicative of an absolute value of a result of subtraction operations wherein bytes of the first operand are subtracted from the corresponding bytes of the second operand.

25. (Original): The byte execution unit as recited in claim 20, wherein in the event the byte instruction information specifies the sum bytes into halfwords operation, the result signal is indicative of sums of values of consecutive bytes of the first and second operands.

26. (Original): The byte execution unit as recited in claim 20, wherein the first compressor unit of one of the byte units is coupled to receive a 32-bit portion A[0:31] of the first operand, and wherein the first plurality of compressor output signals produced by the second compressor unit comprises output signals F2[0:7], F2[8], and F3[0:7], and

wherein the F2[0] signal conveys a carry value resulting from an addition operation A[0]+A[8]+A[16], and wherein the F2[1:8] signal conveys a sum vector, and wherein the F2[8] signal conveys a sum value resulting from an addition operation A[7]+A[15]+A[23]+A[31], and wherein the F3[0:7] signal conveys a carry vector.

27. (Original): The byte execution unit as recited in claim 20, wherein the second compressor unit of one of the byte units is coupled to receive a portion B[0:31] of the second operand, and wherein the second plurality of compressor output signals produced by the second compressor unit comprises output signals F0[0:7], F0[8], and F1[0:7], and wherein the F0[0] signal conveys a carry value resulting from an addition operation B[0]+B[8]+B[16], and wherein the F0[1 :8] signal conveys a sum vector, and wherein the F0[8] signal conveys a sum value resulting from an addition operation B[7]+B[15]+B[23]+B[31], and wherein the F1[0:7] signal conveys a carry vector.

28. (Original): The byte execution unit as recited in claim 20, wherein the adder logic comprises a plurality of 8-bit compound adders.

29. (Currently Amended): A data processing system, comprising:

a memory system comprising a byte instruction, wherein the byte instruction specifies either a count ones in bytes operation, an average bytes operation, an absolute differences of bytes operation, or a sum bytes into halfwords operation; and

a processor coupled to the memory system and configured to fetch and execute instructions from the memory system, wherein the processor comprises:

a byte execution unit coupled to receive byte instruction information ~~and two operands~~, to receive a first operand from a first source register, and to receive a second operand from a second source register, and configured to perform an operation specified by the byte instruction information upon at least one of the ~~two operands~~ first operand or the second operand, thereby producing a result in a destination register,

wherein responsive to a count ones in bytes operation, the byte execution unit counts a number of logical one bits in each byte of at least the first operand and stores each result in a corresponding byte of the destination register;

wherein responsive to an average bytes operation, the byte execution unit averages each byte of the first operand with a corresponding byte of the second operand and stores each result in a corresponding byte of the destination register;

wherein responsive to an absolute differences of bytes operation, the byte execution unit subtracts each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, determines an absolute value of each intermediate result to form a final result, and stores each final result in a corresponding byte of the destination register; and

wherein responsive to a sum bytes into halfwords operation, the byte execution unit sums a number of corresponding one-byte portions of the first operand or the second operand and stores each result in a corresponding halfword of the destination register.

30. (Currently Amended): The data processing system as recited in claim 29, wherein each of the ~~two operands~~ first operand and the second operand comprises a plurality of bits, and wherein the bits of each of the ~~two operands~~ first operand and the second operand are grouped to form a plurality of corresponding 8-bit bytes.

31. (Currently Amended): The data processing system as recited in claim 30, wherein each of the ~~two operands~~ first operand and the second operand comprises 128 bits, and wherein the bits of each of the ~~two operands~~ first operand and the second operand are grouped to form 16 corresponding bytes.

32. (New): The byte execution unit of claim 19, wherein responsive to a count ones in bytes operation, the byte execution unit counts a number of logical one bits in each byte of at least a first operand and stores each result in a corresponding byte of a destination register;

wherein responsive to an average bytes operation, the byte execution unit averages each byte of the first operand with a corresponding byte of a second operand and stores each result in a corresponding byte of the destination register;

wherein responsive to an absolute differences of bytes operation, the byte execution unit subtracts each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, determines an absolute value of each intermediate result to form a final result, and stores each final result in a corresponding byte of the destination register; and

wherein responsive to a sum bytes into halfwords operation, the byte execution unit sums a number of corresponding one-byte portions of the first operand or the second operand and stores each result in a corresponding halfword of the destination register.